PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-237226
(43)Date of publication of application : 09.09.1997

(51)Int.Cl. G06F 12/14

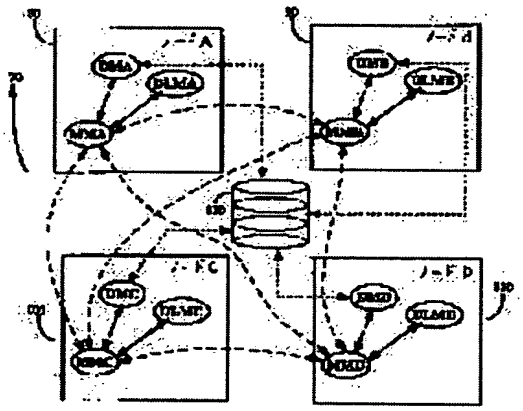
(21)Application number : 08-307544
(22)Date of filing : 05.11.1996

(71)Applicant : SUN MICROSYST INC
(72)Inventor : MATENA VLADIMIR

(30)Priority
Priority number : 95 552316 Priority date : 02.11.1995 Priority country : US

(54) METHOD FOR HIGHLY RELIABLE DISK FENCING IN MULTI-COMPUTER SYSTEM AND DEVICE THEREFOR

(57)Abstract:
PROBLEM TO BE SOLVED: To obtain a method for quickly and accurately separating a resource such as a shared disk in a network system by executing an access request by a peripheral device when 1st and 2nd values respectively expressing the 1st and 2nd constitution of a multi-node system are the same value.
SOLUTION: A new epoch number is generated in each new constitution consisting of a node and a resource in the system 70. Control keys based upon respective epoch numbers are generated and stored in respective resource controllers and nodes 80 to 110. At the time of discriminating the generation of a fault in a certain node, the node is removed from a membership list and a new epoch number and a control key are generated. When an access request is sent from a certain node out of the nodes 80 to 110 to its corresponding resource, the resource controller compares the locally stored control key with the control key stored in the node, and only when both the keys match with each other, the access request is executed.



LEGAL STATUS

[Date of request for examination] 05.11.2003

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

(19)日本国特許庁（ＪＰ）　(12) 公開特許公報（A）　(11)特許出願公開番号

特開平9－237226

(43)公開日　平成9年(1997)9月9日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　12/14	３１０		Ｇ０６Ｆ　12/14	３１０Ａ	

審査請求　未請求　請求項の数2　ＦＤ　（全 10 頁）

(21)出願番号　特願平8－307544

(22)出願日　平成8年(1996)11月5日

(31)優先権主張番号　０８／５５２３１６

(32)優先日　1995年11月2日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　591064003
サン・マイクロシステムズ・インコーポレーテッド
SUN MICROSYSTEMS, INCORPORATED
アメリカ合衆国　94043　カリフォルニア州・マウンテンビュー・ガルシア　アヴェニュウ・2550

(72)発明者　ウラジミール・マテナ
アメリカ合衆国　94061　カリフォルニア州・レッドウッド　シティ・ケントフィールド　アヴェニュ　1322

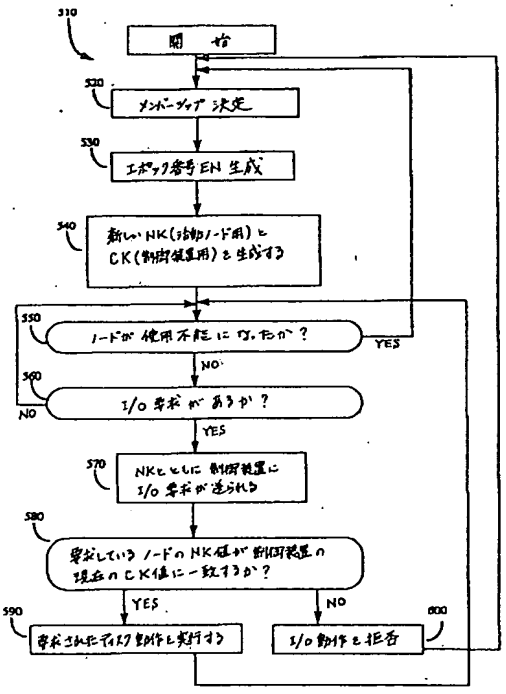
(74)代理人　弁理士　山川　政樹

(54)【発明の名称】　マルチコンピュータ・システムにおける信頼性の高いディスク・フェンシングのための方法および装置

(57)【要約】

【課題】　ネットワーク化システム上の共用ディスクなどの資源の高速かつ確実な分離のための方法および装置を提供する。

【解決手段】　システム上のノードと資源からなる新しい構成ごとに、新しいメンバシップ・リストを生成し、それに基づいて、それが存在する時期と相関関係にあるメンバシップを明確に識別する新しいエポック番号を生成する。エポック番号に基づく制御キーが生成され、システム上の各資源制御装置およびノードで格納される。あるノードが障害が発生したものと識別されると、それはメンバシップ・リストから除去され、新しいエポック番号と制御キーが生成される。ノードが資源に対してアクセス要求を送ると、資源制御装置は、ローカルに格納されたその制御キーと、ノードで格納された制御キー（アクセス要求とともに伝送されたもの）とを比較する。２つのキーが一致した場合のみ、アクセス要求が実行される。



1

【 特許請求の範囲】
【 請求項1 】　マルチノード・システム内のプロセッサベースのノードによる共用周辺装置へのアクセスを防止するための方法において、
(１) マルチノード・システムの第1 の構成を表す第1 の固有の値を周辺装置側で格納するステップと、
(２) マルチノード・システムの第2 の構成を表す第2 の固有の値を含むアクセス要求を、ノードから周辺装置へ送るステップと、
(３) 前記第1 の値と第2 の値が同一であるかどうかを判定するステップと、
(４) 第1 および第2 の値が同一である場合、周辺装置側でアクセス要求を実行するステップとを含むことを特徴とする方法。
【 請求項2 】　周辺資源が制御装置メモリを含む資源制御装置によってマルチノード・システムに結合されており、そのシステムの複数のプロセッサベースのノードのそれぞれが諸機能を実行するように構成されたプログラム・モジュールを格納するノード・メモリに結合されたプロセッサを含むときに、マルチノード・システム内のノードによる少なくとも１つの共用周辺資源へのアクセスを防止する装置において、
システムのメンバシップが変更された時期を少なくとも含む所定の時期にマルチノード・システム上の前記資源を含むノードのメンバシップ・リストを判定するように構成されたメンバシップ・モニタ・モジュールと、
資源が障害状態になったときを判定するように構成され、資源の障害を前記メンバシップ・モニタに連絡して、新しいメンバシップ・リストを生成するようにメンバシップ・モニタに指示する資源マネージャ・モジュールと、
前記新しいメンバシップ・リストに基づいて固有の値を生成し、マルチノード・システム上の各ノード側で前記固有の値をローカルに格納するように構成された構成値モジュールと、
前記制御装置メモリ側で格納され、前記要求側ノードでローカルに格納された固有の値が前記資源制御装置で格納された固有の値と等しくないときに少なくとも１つの前記要求側ノードによる前記資源へのアクセス要求をブロックするように構成されたアクセス制御モジュールとを含むことを特徴とする装置。
【 発明の詳細な説明】
【０００１】
【 発明の属する技術分野】本発明は、複数のコンピュータ(ノード) が共用ディスクに対して並行アクセス可能なマルチコンピュータ・システム、たとえば、クラスタにおける共用ディスクの信頼性の高いディスク・フェンシング(disk fencing) のためのシステムに関する。具体的には、このシステムは、共用アクセス・ディスクを備えた高可用性のシステムに対処するものである。

2

【０００２】
【 従来の技術】クラスタ化コンピュータ・システムでは、他のノードが従う任意の定義済み基準により、所与のノードで「 障害が発生する」、すなわち、それが使用不能になることがある。たとえば、所与のノードが任意の所定の時間より短時間の間に要求に応答できなかった可能性が考えられる。したがって、異常にゆっくり実行しているノードは障害が発生したと見なされる可能性があり、それに応じて残りのノードが応答することになる。
【０００３】１つのノード (または複数のノード) で障害が発生すると、残りのノードはシステム再構成を実行して、障害が発生したノード (複数も可) をシステムから除去しなければならず、その場合、残りのノードは障害が発生したノード (複数も可) が提供していたサービスを提供することが好ましい。
【０００４】障害が発生したノードをできるだけ迅速に共用ディスクから分離することが重要である。そうでない場合、システム再構成が完了するまでに障害が発生した(またはゆっくり実行している) ノードが分離されないと、そのノードが共用ディスクに対して読み書き要求を出し続ける可能性があり、それにより、共用ディスク上のデータを破壊する恐れがある。
【０００５】この種の問題に対処するために、ディスク・フェンシングプロトコルが開発されている。たとえば、ＶＡＸクラスタ・システムでは、「 デッドマン・ブレーキ」機構を使用する。参照により本明細書に組み込まれるＤａｖｉｓ，Ｒ．Ｊ．のＶＡＸ　ｃｌｕｓｔｅｒ　Ｐｒｉｎｃｉｐｌｅｓ（Ｄｉｇｉｔａｌ　Ｐｒｅｓｓ，１９９３）を参照されたい。ＶＡＸクラスタ・システムでは、障害が発生したノードが新しい構成から分離され、新しい構成内のノードは、ディスクへのアクセスが許可されるまでの所与の所定のタイムアウト期間中、待機しなければならない。分離したノード上のデッドマン・ブレーキ機構により、タイムアウト期間の終了まで、分離したノードが「 アイドル」状態になることが保証される。
【０００６】ＶＡＸクラスタ・システム内の分離したノード上のデッドマン・ブレーキ機構は、ハードウェアとソフトウェアの両方を含む。分離したノード上のソフトウェアは、共用ディスクとクラスタ相互接続との間に結合されるクラスタ相互接続アダプタ (ＣＩ) に対して、そのノードが「 健全」であることを定期的に通知しなければならない。このソフトウェアは、そのノードが新しい構成の一部ではないことを限られた時間内に検出することができる。このような状態が検出されると、ソフトウェアはディスク入出力をブロックし、その結果、障害が発生したノードによる共用ディスクへのすべてのアクセスを防止するソフトウェア「 フェンス」がセットアップされる。このソフトウェア・フェンスがもたらす欠点

3

は、ソフトウェアが信頼性の高いものでなければならないことであり、「フェンス」ソフトウェアの障害（またはバグ）が発生すると、その結果、表面上分離したノードによる共用ディスクへのアクセスをブロックできなくなる。

【０００７】ソフトウェアの実行速度が遅すぎ、かつ適切にソフトウェア・フェンスがセットアップされない場合、ＣＩハードウェアはそのノードを相互接続から遮断し、それにより、ハードウェア・フェンス、すなわち、障害が発生したノードが共用ディスクにアクセスできないようにするハードウェア障害物をセットアップする。このハードウェア・フェンスは、ＣＩホスト・アダプタ上の健全タイマにより実現される。ソフトウェアは、そのソフトウェアが「健全」であることをＣＩハードウェアに定期的に通知しなければならない。所与のタイムアウト期間内にこれを行わないと、ＣＩ内の健全タイマが起動されることになる。これが「デッドマン・ブレーキ」機構である。

【０００８】このノード分離システムには、上記以外に次のような欠点がある。

・ハードウェア・フェンスを実現するために内部タイマを使用する相互接続アダプタが必要である。

・ノードとディスクとの間の相互接続がスイッチまたはその他のバッファ装置を含む場合、この解決策は機能しない。分離したノードからのディスク要求は本来、このようなスイッチまたはバッファによって遅延され、新しい構成がすでにディスクにアクセスした後にディスクに送られるはずである。このような要求の遅延によって、ファイルまたはデータベースが破壊される恐れがある。

・様々なタイムアウト値によっては、新しい構成のメンバがディスクにアクセスできるようになるまで待機しなければならない時間が長くなりすぎる場合があり、その結果、システム全体のパフォーマンスが低下し、高可用性の原則に反することになる。

【０００９】アーキテクチャ・レベルで見た場合、上記のノード分離方法の重大な欠点は、そのノードが端末間特性を備えていないことであり、フェンスがディスク制御装置上ではなく、ノード上にセットアップされることである。

【００１０】ディスク制御装置で障害が発生したディスクの分離を迅速にセットアップしながら高可用性を提示するシステムを用意すると、有利であると思われる。

【００１１】UNIXベースのその他のクラスタ化システムでは、クラスタ化ノードの不要なサブセットが共用ディスクにアクセスできないようにするために、ＳＣＳＩ（small computer systems ubterface 小型コンピュータ・システム・インタフェース）の「ディスク予約」を使用する。たとえば、情報システムに関するＡＮＳＩ ＳＣＳＩ －２規格案（１９９０年3月9日付け、Global Engineering Document

4

s が配布）を参照されたい。これは参照により本明細書に組み込まれるものである。ディスク予約にはいくつかの欠点がある。たとえば、ディスク予約プロトコルは、２つのノードを有するシステムにしか適用できない。というのは、ノードが1つだけの場合、一度に1枚ずつディスクを予約できる（すなわち、他のどのノードも同時にそのディスクにアクセスすることができない）からである。もう１つは、ＳＣＳＩシステムでは、ＳＣＳＩバス・リセット動作によりディスク予約が除去され、ソフトウェア・ディスク・ドライバがいつでもＳＣＳＩバス・リセットを出すことができることである。したがって、ＳＣＳＩのディスク予約は、信頼性の高いディスク・フェンシング技法ではない。

【００１２】もう１つのノード分離方法は「毒薬」を含む。すなわち、再構成中にシステムから１つのノードが除去されると、残りのノードの1つが障害が発生したノードに対して「毒薬」すなわち停止要求を送るのである。障害が発生したノードが活動状態にある（たとえば、ゆっくり実行している）場合、そのノードは毒薬を摂取し、所定の時間内にアイドル状態になる。

【００１３】この毒薬は、障害が発生したノードのホスト・アダプタ・カードまたは障害が発生したノード上の割込みハンドラのいずれかによって処理される。これがホスト・アダプタ・カードによって処理される場合、この方法を実現するためにシステムは特別に設計したホスト・アダプタ・カードを必要とするという欠点がもたらされる。障害が発生したノード上の割込みハンドラによって処理される場合は、ノード分離が信頼できなくなるという欠点がある。たとえば、上記のＶＡＸクラスタの場合のように、ノードのソフトウェア自体が信頼できないものになる可能性があり、タイムアウト遅延が発生し、やはり共用ディスクではなくノードで分離が行われる。

【００１４】

【発明が解決しようとする課題】したがって、分離したノードが共用ディスクにアクセスするのを迅速かつ確実にブロックし、ディスク・アクセスへの防止をするのに分離したノードに依存しない機構を使用して、共用ディスクへのアクセスをそのディスク側で防止するシステムが要求されている。

【００１５】

【課題を解決するための手段】本発明は、共用ディスクなどの入出力装置を含む、障害が発生した資源を迅速かつ確実に分離するための方法および装置で、コンピュータ・システムまたはネットワーク上のほぼすべての共用資源に適用可能である。本発明のシステムは、すべての活動状態の共用資源のメンバシップ・リストを維持し、資源が追加されるかまたは障害が発生した場合（その結果、機能的に除去しなければならない場合）など、新しい構成のたびに、システムは、その時点でその構成を明

5

確に識別する新しいエポック番号またはその他の値を生成する。したがって、異なる時点で生成した同一のメンバシップは、特に、中間に異なるメンバシップ・セットが発生した場合には、エポック番号が異なることになる。

【００１６】新しいエポック番号が生成されるたびに、それから制御キー値が導出され、システム内のノードに送られる。それぞれのノードは、それ専用のノード・キーとしてその制御キーをローカルに格納する。資源用の制御装置（ディスク制御装置など）も制御キーをローカルに格納する。その後、共用資源アクセス要求が資源制御装置に送られると、それとともに必ずノード・キーが送られる。次に制御装置は、そのノード・キーが制御装置が格納しているバージョンの制御キーと一致するかどうかを検査し、２つのキーが一致する場合のみ、資源アクセス要求を許可する。

【００１７】資源ファイルで障害が発生した場合、たとえば、資源ファイルが所定の期間内に要求に応答しない場合（ハードウェアまたはソフトウェア欠陥の可能性を示す）、システムのメンバシップは新規と判定され、障害が発生した資源が除去される。新しいエポック番号が生成され、そこから新しい制御キーが生成され、システム上のすべての資源制御装置およびノードに伝送される。新しい制御キーの生成された後でアクセス要求が資源制御装置に到達した場合、そのアクセス要求には、現行制御キーとは異なるノード・キーが付けられ、したがって、その要求は実行されないことになる。これは、ノードが現行メンバシップ・セットに含まれない資源にアクセス要求を出すのを防止することと結合すると、処理するためにはすべてのノード要求が現行制御キー（およびその結果のメンバシップ）情報を有していなければならないと要求することにより、障害が発生した資源が迅速にアクセスから除去されることになる。

【００１８】それぞれのノードは、本発明の諸機能を実行するためにプログラム・モジュール、たとえば、ディスク（または資源）マネージャ・モジュール、分散ロック・マネージャ・モジュール、メンバシップ・モジュールを格納している。このようなモジュールの分散によって、どのノードも、障害が発生したものとして資源を識別し、残りのノードにそれを連絡し、新しいメンバシップ・リスト、エポック番号、制御キーを生成できるようになる。

【００１９】したがって、上記のシステムは、障害が発生した資源のハードウェアまたはソフトウェアに依存せず、高速の端末間（すなわち、資源側での）資源分離を可能にする。

【００２０】

【発明の実施の形態】本発明のシステムは、一般に、複数のノード２０～４０（この例ではノード１～３）と１組または複数組の共用ディスク５０とを含む、図1に示

6

すシステム１０などのクラスタ化システムに適用可能である。ノード２０～４０のそれぞれは、１つまたは複数のプロセッサを有し、メモリ、大容量記憶装置、ユーザ入出力装置（モニタ、キーボード、マウスなど）、その他の従来のコンピュータ・システム要素（図1にすべて示されているわけではない）を含む、従来のプロセッサベースのシステムにすることができ、クラスタ化環境で動作するように構成することができる。

【００２１】ディスク５０はディスク制御装置６０によりアクセスされ制御されるが、この制御装置は従来のディスク制御装置のハードウェアとソフトウェアを含むことができ、以下に記載する特徴に加え、ディスク制御機能を実行するためにプロセッサとメモリ（個別に図示せず）とを含む。

【００２２】一般に、本発明のシステムは、ノード２０～４０およびディスク制御装置のメモリに格納されたソフトウェア・モジュールによって実現することができる。このソフトウェア・モジュールは、本発明のディスク・フェンシングシステムを実現するための適当な要素に関する以下の教示に基づいて、従来のソフトウェア・エンジニアリングによって構築することができる。したがって、一般に以下の説明では、記載する各機能は、ノードまたは資源（たとえば、ディスク）制御装置のいずれか該当する方に格納された個別のプログラム・モジュールによって実現することができ、また、このような機能のいくつかは、単一の多目的モジュールによって効果的に実現することもできる。

【００２３】図2は、本発明を実現するクラスタ化システム７０をより詳細に示している。システム７０は、４つのノード８０～１１０（ノードＡ～Ｄ）と、少なくとも１つの共用ディスク・システム１２０とを含む。ノード８０～１１０は、従来のどのようなクラスタ・ノード（ノード２０～４０またはその他の適切なクラスタ・ノードのように、ワークステーション、パーソナル・コンピュータ、その他のプロセッサベースのシステムなど）にすることもでき、ディスク・システムは、図1に関連して述べたように、ディスク・システム５０を含む、適切な共用ディスク・アセンブリにすることができる。

【００２４】各ノード８０～１１０は、少なくとも以下のソフトウェア・モジュールを含む。すなわち、ディスク・マネージャ（ＤＭ）、任意の分散ロック・マネージャ（ＤＬＭ）、メンバシップ・モニタ（ＭＭ）である。これらのモジュールは、大部分はクラスタ化コンピューティングの分野に従来からあるものでよく、本発明の特徴を実現するために必要な修正を加えたものにすることができる。４つのMMモジュールMMA～MMDは、図２に示すように互いに連絡した状態で接続されており、ディスク・マネージャ・モジュールＤＭＡ～ＤＭＤのそれぞれはディスク・システム１２０のディスク制御装置（個別に図示せず）に結合されている。

7

【００２５】従来のクラスタ化システムのノードは、上記のＶＡＸ　ｃｌｕｓｔｅｒ　Ｐｒｉｎｃｉｐｌｅｓに記載されているような「メンバシップ・プロトコル」に関与する。メンバシップ・プロトコルは、障害の認知によって所与のノードが脱落したときに新しい構成を形成するノードの組について合意を確立するために使用するものである。メンバシップ・プロトコルを使用すると、（ａ）システムの現行メンバであると見なされるノードのサブセットと、（ｂ）システムの現行状況を反映した「エポック番号」（ＥＮ）とを含む出力が得られる。ＥＮに代わるものとしては、所与の時間のシステムの状況を明確に反映する時間値または状況値がある。本システムでは、このようなメンバシップ・プロトコルを使用することができる。

【００２６】メンバシップ・プロトコルにより、メンバシップ・セットが変わると、新しい固有のエポック番号が必ず生成され、新しいメンバシップ・セットと関連付けられる。たとえば、システムが（図2のように）４つのノードＡ～Ｄからなるメンバシップで始まり、エポック番号１００が現行構成に割り当てられている場合、これは＜Ａ，Ｂ，Ｃ，Ｄ；＃１００＞または＜ＭＥＭ＝Ａ，Ｂ，Ｃ，Ｄ；ＥＮ＝１００＞として表すことができる。ただし、ＭＥＭは「メンバシップ」を表す。これは、図3（ａ）に表されている構成であり、４つのノードすべてが活動状態で、クラスタ内の関与ノードになっている。

【００２７】ノードＤがクラッシュするかまたは誤動作しているものとして検出された場合、新しいメンバシップは＜ＭＥＭ＝Ａ，Ｂ，Ｃ；ＥＮ＝１０１＞になる。すなわち、ノードＤがメンバシップ・リストから除去され、エポック番号が１０１に増分されて、Ｄがほんの最近までメンバーだったというエポックが終了したことを示す。新しいメンバシップに関与するすべてのノードは新しいメンバシップ・リストと新しいエポック番号を格納するが、障害が発生したノードＤ（および障害が発生したその他のノード）は古いメンバシップ・リストと古いエポック番号を維持する。これは図3（ｂ）に示す通りであり、ノードＡ～Ｃのメモリはいずれも＜ＭＥＭ＝Ａ，Ｂ，Ｃ；ＥＮ＝１０１＞を格納しているが、障害が発生し分離されたノードＤは＜ＭＥＭ＝Ａ，Ｂ，Ｃ，Ｄ；ＥＮ＝１００＞を格納している。

【００２８】本発明では、この事実、すなわち、現行情報は活動状態のノードによって格納され、時代遅れの情報は分離したノード（複数も可）によって格納されるということを利用して、ディスク・フェンシングを達成する。これは、ノードと共用ディスク・システムの制御装置（たとえば、ディスク制御装置の揮発性メモリ）によって格納された「制御キー」（ＣＫ）変数の値を使用することによって行われる。

【００２９】図4は、ノード４１０～４４０と、ディスク４５２～４５６（システム４５０）と４６２～４６６（システム４６０）を含む2つの共用ディスク・システム４５０～４６０とを含む、４ノード・クラスタ化・システム４００のブロック図である。ディスク・システム４５０および４６０はそれらとクラスタ相互接続４９０との間に接続されたそれぞれ用のディスク制御装置４７０および４８０によって制御される。この相互接続はディスク制御装置とノードとの間に設けられている。

8

【００３０】ノード４１０～４４０は、前述のようにプロセッサベースのシステムにすることができ、ディスク制御装置も前述の通りであるので、ノード、共用ディスク・システム（制御装置あり）、クラスタ相互接続は、当技術分野で従来のものにここに記載する特徴を追加したものにすることができる。

【００３１】各ノードは、「ノード・キー」（ＮＫ）変数とメンバシップ情報の両方を格納する。ＮＫ値は、方法1～3として以下に記載するいくつかの代替機能の1つによって、現行メンバシップから計算される。図4は、いずれかのノードで障害が発生し、そのノードがメンバシップ・セットから除外された場合に、そのノードが残りのノードとは異なるＣＫ番号を有する可能性があることを考慮に入れた汎用状況を示している。

【００３２】しかし、原則として、すべてのノードが活動状態の場合、それぞれの格納ＮＫ値とディスク制御装置で格納されたＣＫ値はすべて等しくなる。

【００３３】ノード・キー値および制御キー値を使用する

ノード／ディスク制御装置の動作

ディスク制御装置にアクセスするためのノードによる各読み書き要求はＮＫ値を含む。すなわち、ノードが共用ディスクへの読取りまたは書込みアクセスを要求する場合、必ずその要求の一部としてＮＫ値が渡される。したがって、このように読み書き要求にＮＫ値を含めることがノードと制御装置（複数も可）との間のプロトコルの一部を構成する。

【００３４】ノードとディスク制御装置との間のプロトコルは、制御装置上でＣＫ値を操作する２つの動作も含む。すなわち、現行ＣＫ値を読み取るためのＧｅｔ　Ｋｅｙと、ＣＫの値を新しい値に設定するＳｅｔ　Ｋｅｙである。Ｇｅｔ　Ｋｅｙでは、ＮＫ値、ＣＫ値、ＥＮ値を供給する必要はないが、Ｓｅｔ　Ｋｅｙプロトコルでは、ＮＫ値を入力として使用し、さらに制御装置が採用すべき新しいＣＫ値「ｎｅｗ．ＣＫ」を供給する。

【００３５】上記4通りの要求とそれぞれの入出力引数は、次のように表し、まとめることができる。

Ｒｅａｄ（ＮＫ，．．．）

Ｗｒｉｔｅ（ＮＫ，．．．）

Ｇｅｔ　Ｋｅｙ（．．．）

Ｓｅｔ　Ｋｅｙ（ＮＫ，ｎｅｗ．ＣＫ）

【００３６】Ｇｅｔ　Ｋｅｙ（．．．）動作は、ＣＫの現

行値を返す。この動作が制御装置によって拒否されることはない。

【００３７】Ｓｅｔ Ｋｅｙ（ＮＫ，ｎｅｗ．ＣＫ）動作は、まず、要求のＮＫフィールドが制御装置内の現行ＣＫ値と一致するかどうかを検査する。一致する場合、制御装置内のＣＫ値は（Ｓｅｔ Ｋｅｙ要求の）「ｎｅｗ．ＣＫ」フィールド内の値に等しくなるように設定される。要求側ノードからのＮＫが制御装置に格納された現行ＣＫ値と一致しない場合、この動作が拒否され、要求側ノードにエラー表示が送られる。

【００３８】パケット内のＮＫフィールドがＣＫの現行値と一致する場合のみディスクにアクセスするために、Ｒｅａｄ（ＮＫ，．．．）動作とＷｒｉｔｅ（ＮＫ，．．．）動作が許可される。それ以外の場合、この動作は制御装置によって拒否され、要求側ノードにエラー表示が送られる。

【００３９】制御装置を始動したときに、ＣＫ値が0に初期設定されることが好ましい。

【００４０】ノードの障害時の手順

障害が発生した1つまたは複数のノードがシステムから除去されるためにメンバシップが変わる場合、残りのノードは、（後述するように）新しいメンバシップ情報からＣＫの新しい値を計算する。ノードの1つは、Ｓｅｔ Ｋｅｙ（ＮＫ，ｎｅｗ．ＣＫ）動作を使用してディスク制御装置に新しいＣＫ値を連絡する。新しいＣＫ値が設定された後、新しい構成のすべてのメンバ（活動）ノードがそれぞれのＮＫ値をこの新しいＣＫ値に設定する。

【００４１】ノードが新しい構成の一部ではない（たとえば、障害が発生したノード）場合、そのＮＫを変更することができない。このようなノードがディスクの読取りまたは書込みを試みると、制御装置は、新しいＣＫ値と古いＮＫ値との不一致を検出する。

【００４２】ノードを始動すると、そのＮＫが0値に初期設定される。

【００４３】制御キー（ＣＫ）の値を計算するための手順

制御キーＣＫはいくつかの方法で設定することができる。選択した計算は、少なくとも制御装置側で格納または搭載されるソフトウェアまたはファームウェアに反映される。一般に、ＣＫ値の計算は、次のようなメンバシップ情報を考慮に入れる必要がある。

ＣＫ＝ｆｕｎｃ（ＭＥＭ，ＥＮ）

式中、ＭＥＭは活動状態のメンバシップ・リストに関する情報を含み、ＥＮはエポック番号である。

【００４４】方法1．ＣＫ値が新しいメンバシップ・セット（複数ノードの符号化セット）のリストとエポック番号の両方を明示的に含むことが理想的であると思われる。しかし、ノードの数が大きい場合、これは不要であると思われる。というのは、ＣＫの値が各ノードの情報の少なくとも１ビット分を含む必要があるはずだからである。すなわち、４ノード構成では、少なくとも４ビット・シーケンスＢＢＢＢ（ただし、Ｂ＝0または1）を使用する必要があり、各ビットＢは、所与の関連ノードが活動状態か非活動状態（障害が発生した状態）かを示すはずである。さらに、エポック番号ＥＮのためにいくつかのビットが必要であり、そのため、変数ＣＫの全長が非常に長くなる可能性がある。

【００４５】以下の方法2と方法3は、ＣＫ値を計算するときのメンバシップ情報を圧縮するために設計されたものである。

【００４６】方法2では、エポック番号ＥＮのみを使用し、メンバシップ・リストＭＥＭを無視する。例えば、ＣＫ値はエポック番号E-Nと等しくなるように設定される。

【００４７】メンバシップ・プロトコルが（たとえば、多数決により）ネットワークの区分化を防止する場合、方法2が最も実用的である。ハードウェア障害の場合など、メンバシップの区分化が許される場合、クラスタの実際のメンバシップを反映せずにＣＫ値を使用すると、区分のいずれかの側のノード間で競合が発生する可能性がある。

【００４８】方法3は、区分に関する方法2の課題を解決するものである。この方法では、新しい構成の最高ノードのＩＤによってＣＫ値が符号化される。たとえば、ＣＫ値は、ノードＩＤ（最高ノードに割り当てられる番号）とエポック番号の連結にすることができる。この方法は、メンバシップ・モニタ自体がネットワークの区分化を防止しない場合でも安全なディスク・フェンシングを可能にする。というのは、所与の区分内の最高ノードの番号が別の区分のそれとは異なるからであり、このため、各種サブクラスタ用のＥＮが偶然同じになった場合でも、各種区分内のノードからの要求間で競合が発生することがない。

【００４９】上記の方法のうち、ノードの数が少ない場合は方法1が好ましい。というのは、クラスタ化システムの状態に関する最も明示的な情報が含まれるからである。しかし、ノードの数が多い場合は方法3が好ましくなる。システムがネットワークの区分化を防止する場合は、方法2が適当である。

【００５０】本発明の方法

前記の構造および機能ならびにそれらを実現するための適切なモジュールの場合、本発明のディスク・フェンシングシステムは、図5の流れ図に示す方法５１０に従うことによって実施される。ボックス（ステップ）５２０では、クラスタ化システムのメンバシップが従来通りに決定され、メンバシップ・セット（またはリスト）の値がＭＥＭの値として格納される。エポック番号ＥＮ（またはその他の固有の状態ＩＤ）はボックス５３０で生成される。この2つの機能はメンバシップ・モニタ（ＭＭ）モジュールによって実行されるが、このモジュール

は、どのノードがシステム内に存在するかを判定し、ＥＮの値をその構成に割り当てるために、メンバ・ノード間で実現される。このようにMMモジュールを使用するシステムの一例としては、出願人であるＳｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，Ｉｎｃ．のＳｐａｒｃＣｌｕｓｔｅｒ　ＰＤＢ（パラレル・データベース）がある。

【００５１】現行システムでは、所与のメッセージまたはデータ・パケットが古くなっているかどうかをノードが判定できるように、エポック番号が使用される。すなわち、エポック番号が時代遅れになっている場合、メッセージは、そのクラスタの以前の別の構成中に作成されたものであると認識される。たとえば、Ｔ．Ｍａｎｎ他の「Ａｎ　Ａｌｇｏｒｉｔｈｍ　ｆｏｒ　Ｄａｔａ　Ｒｅｐｌｉｃａｔｉｏｎ」という１９８９年6月のＤＥＣ　ＳＲＣリサーチ・レポートを参照されたい。これは参照により本明細書に組み込まれるものであり、同書には、分散システム内でファイル複製にスタンプする際に使用するものとしてエポック番号が記載されている。

【００５２】本システムでは、まったく新しいやり方でエポック番号を使用するが、これはエポック番号の従来システムの使い方とは無関係である。Ｓｕｎ　Ｍｉｃｒｏｓｙｓｔｅｍｓ，Ｉｎｃ．のシステムにおけるクラスタ・メンバシップ・モニタの好ましい使い方の例については、本明細書に付属の付録Ａを参照されたい。その場合の再構成シーケンス番号はエポック番号と類似のものである。したがって、本発明により積年の問題が解決されるという明確な利点がもたらされる。すなわち、新しい出力を生成してプロセスを制御するのに新しい手順を必要とせずに、障害が発生したノードを迅速かつ確実にクラスタ・メンバシップから除去し、そのノードが共用ディスクにアクセスし続けるのを防止する。また、すでに生成されたタイプの情報を本発明によるモジュールとともに使用すると、所望の機能を実施することができ、その結果、信頼性の高い高可用性のシステムが得られる。

【００５３】ボックス５４０に移行すると、上記の方法１～3のいずれか1つまたは他の適当な方法により、ノード・キーＮＫ（活動ノード用）と制御キーＣＫが生成される。

【００５４】ボックス５５０では、ノードが使用不能になっているかどうかが判定される。このステップはほぼ連続して（または、少なくとも比較的高い頻度、たとえば、入出力要求の頻度より高い頻度で）実行される。たとえば、ほぼいつでも、所与のノードは、他のノードが要求に応答するための許容時間を超えていることを判定し、後者のノードで障害が発生しているのでそれをクラスタのメンバシップ・セットから除去する必要があると決定することができる。したがって、ボックス５５０のステップは、この方法の実行中のほぼどこでも行うことができる。

【００５５】ボックス５６０は、クラスタに接続されたノードの1つが入出力要求（ディスク・アクセス要求など）を生成するという事象を表している。そのような場合、ボックス５７０では、要求側ノードからのＮＫの現行値が入出力アクセス要求とともに送られ、ボックス５８０では、これが制御装置によって格納されたＣＫの値と一致するかどうかが判定される。一致しない場合、方法はステップ６００に移行し、そこで要求が拒否され（アクションなしで単に制御装置によって落とされることを意味する場合もある）、方法はボックス５２０に戻る。

【００５６】ノードのＮＫ値が制御装置のＣＫ値と一致する場合、ボックス５９０で要求が実行される。

【００５７】あるノードで障害が発生した場合、方法はボックス５５０からボックス５２０に戻り、そこで障害が発生したノードが従来通りメンバシップ・セットから除去され、その結果、ＭＥＭの値がこれを反映するように変化する。この時点で、新しいエポック番号ＥＮが生成され（ボックス５３０）、格納されて、新たに改訂したメンバシップ・リストを反映する。さらに、ボックス５４０では、新しい制御キー値ＣＫが生成され、活動ノードのＮＫ値が新しいＣＫ値の値を獲得し、方法はもう一度ボックス５５０～５６０に移行してさらにディスク・アクセスが行われる。

【００５８】上記により、クラスタ化システム内の所与のノードの障害の結果、クラスタ・メンバシップからそのノードが除去されるとともに、重要なことに、障害が発生したノードが共用ディスクに対してさらにディスク・アクセスするのを確実に防止することが分かるだろう。障害が発生したノードを共用ディスク・アクセスから無効化することは、障害が発生したノードのハードウェアまたはソフトウェアのいずれにも依存せずに適切に機能するが、むしろ、障害が発生したノードとはまったく無関係である。

【００５９】ＣＫ値はディスク制御装置に格納され、障害が発生したノードが共用ディスク・アクセスを獲得するのを防止するためにアクセス制御モジュールによって使用されるので、本発明のディスク・フェンシングシステムは、ディスク管理ソフトウェア自体と同程度に信頼できるものである。したがって、クラスタ化システムは、障害が発生したノードを迅速かつ確実に除去することができ、その共用ディスク上に格納されているデータの保全性を損なうという危険は最小限になる。

【００６０】本発明には、従来のシステムに比べ、その端末間特性によってそれがディスク相互接続ネットワークまたはバス構成とは無関係になるという重要な利点がある。したがって、エポック番号またはその他の固有の状況値を決定する際にノード構成だけが考慮される。すなわち、いかなる低レベル機構（転送機構など）とは無関係である。

13

【００６１】本発明のシステムは、マルチプロセッサ・システム内の複数のノードによってアクセスされる他の周辺装置にも適用可能であることに留意されたい。たとえば、上記の共用ディスクの代わりに、他の入出力装置またはメモリ装置を使用することもできる。また、ディスク制御装置４７０および４８０に対応する制御装置が使用され、分離動作を実行するためのソフトウェア・モジュールを装備しているはずである。

【００６２】さらに、クラスタのメンバであるノード、すなわち、プロセッサベースのシステムは、様々なプロセッサベースの装置のいずれでもよく、特に、パーソナル・コンピュータやワークステーションにする必要はないが、共用ディスクなどの周辺装置にアクセス要求を出すことができる他のプロセッサ駆動装置にすることもできる。

【図面の簡単な説明】

【図1】　１組の共用ディスクへのアクセスが設けられた複数のノードを示す最上位ブロック図である。

14

【図2】　図1のそれと同様のシステムのより詳細なブロック図であるが、ディスク・フェンシングを達成するためにやりとりする本発明のシステムの諸要素を示す図である。

【図3】　ノードＤが使用不能の場合に再構成の前後の図2または図3の各ノードの構造の諸要素を示す図である。

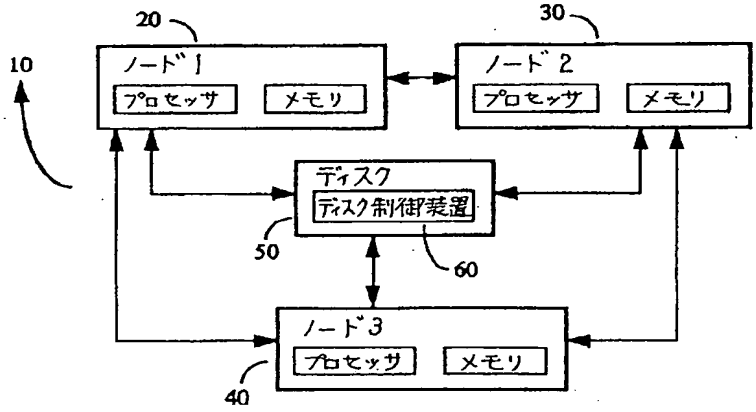
【図4】　ノードが複数組の共用ディスクにアクセスする場合の本発明のシステムのブロック図である。

【図5】　本発明の方法を示す流れ図である。

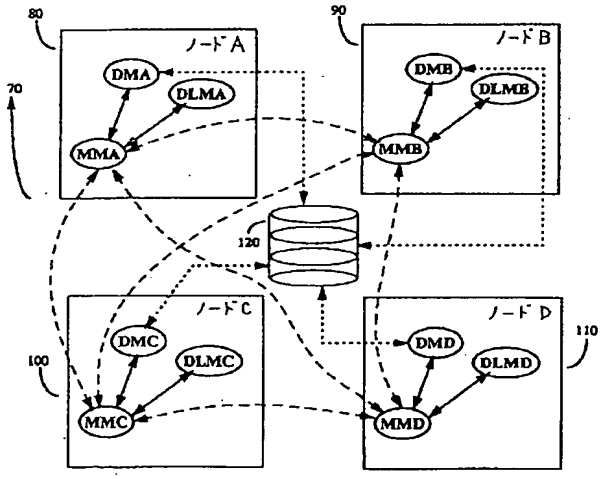
【符号の説明】

１０　システム
２０　ノード
３０　ノード
４０　ノード
５０　共用ディスク
６０　ディスク制御装置

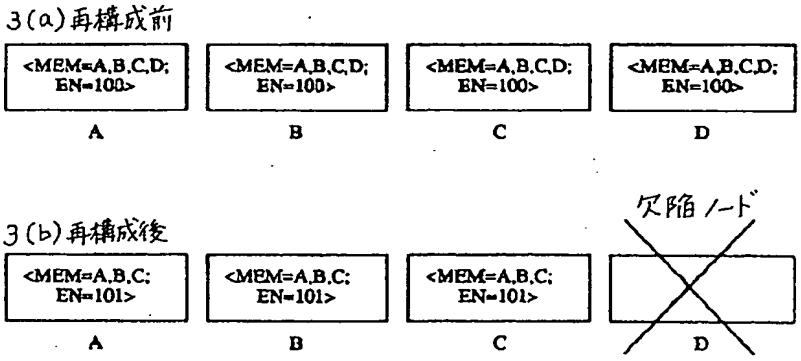
【図1】



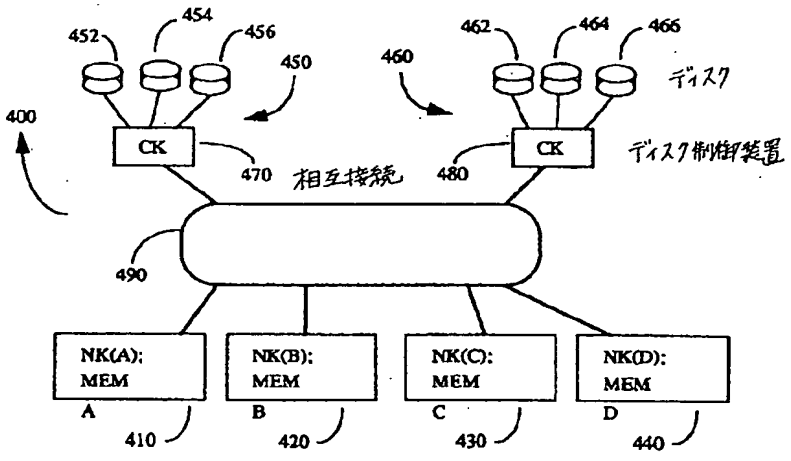
【図2】



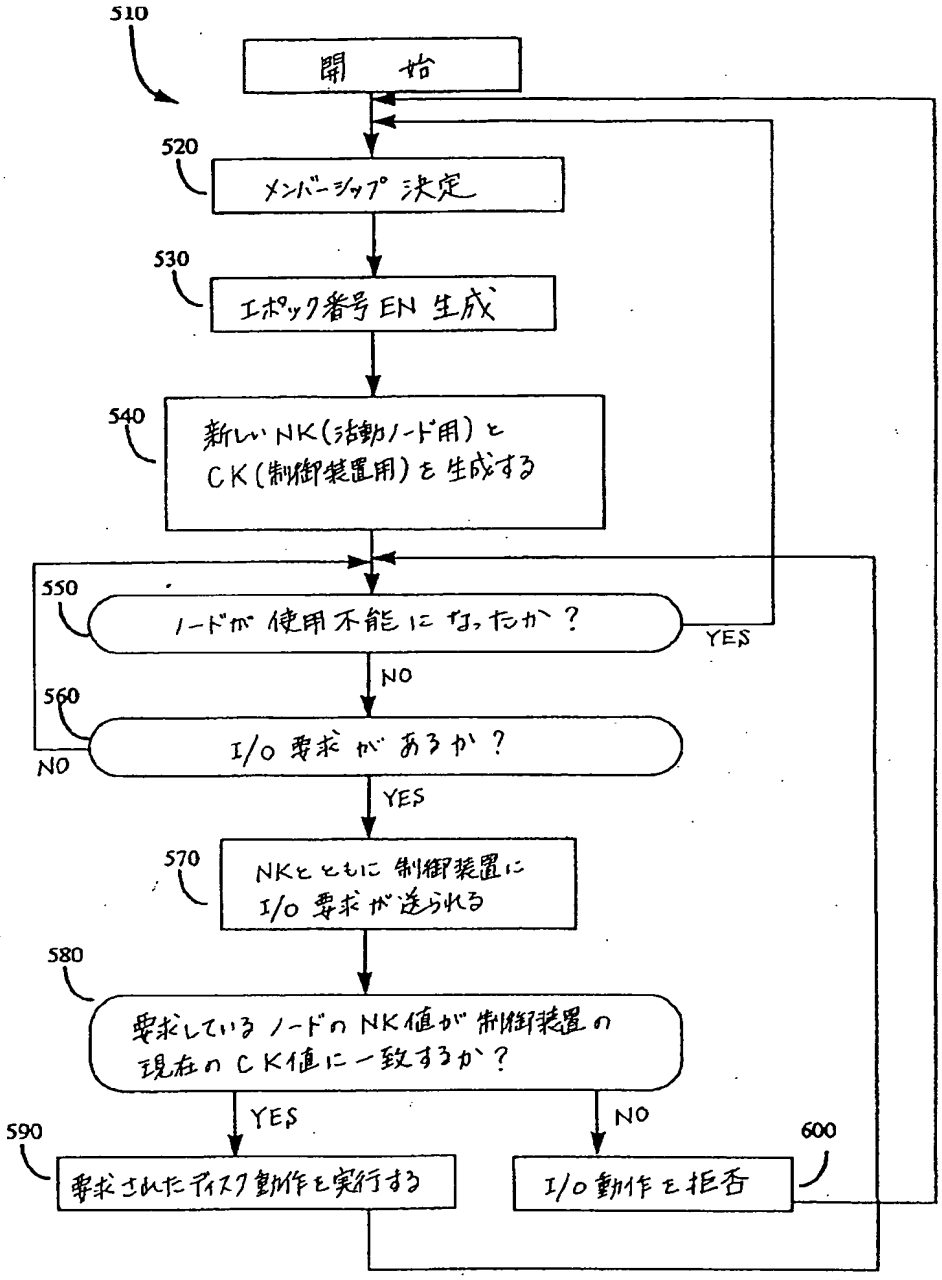
【 図3 】


3(a)再構成前
<MEM=A,B,C,D;
EN=100>
A
B
C
D
3(b)再構成後
欠陥ノード
<MEM=A,B,C;
EN=101>
A
B
C
D

【 図4 】


452
454
456
450
460
462
464
466
ディスク
400
CK
470
相互接続
480
CK
ディスク制御装置
490
NK(A):
MEM
A
410
NK(B):
MEM
B
420
NK(C):
MEM
C
430
NK(D):
MEM
D
440

【 図5 】


510
開　始
520
メンバーシップ 決定
530
エポック番号 EN 生成
540
新しい NK(活動ノード用) と
CK(制御装置用) を生成する
550
ノードが 使用不能 になったか？
YES
NO
560
I/O 要求 があるか？
NO
YES
570
NKとともに 制御装置に
I/O 要求が送られる
580
要求している ノードの NK値が 制御装置の
現在の CK値に一致するか？
YES
NO
590
要求された ディスク動作を実行する
600
I/O 動作を拒否